

שם הפרויקט: Uniapp

שם התלמיד: אופק שטרייכר



קרית החינוך השש שנתית
ע"ש חיים בר-לב
מקיף ח
סגל מוסד 441287



תיק פרויקט

שם התלמיד: אופק שטרייכר

תז של התלמיד: 324826965

שם בית הספר ועיר: מקיף עירוני ח ראשון לציון

שם המנחה: יעקב שוצמן

שם הפרויקט: Uniapp

מועד הגשה: יוני 2020

תוכן עניינים:

מבוא.....	3
מבנה.....	
מדריך למשתמש.....	

מבוא:

ספר הפרויקט יכיל בתוכו מספר דברים שיעזרו לך להבין את הפרויקט שכתבתי במסגרת השתתפותי במגמת הנדסת תוכנה.

ראשית, התוכנה שכתבתי נקראת Uniapp. הרקע לשם הוא שתחילה כאשר כתבתי את התוכנה תכננתי ליישם אותה על אוניברסיטאות, אך ככל שהזמן התקדם החלטתי לשנות אותה לבתי ספר.

מטרת התוכנה היא מערכת תקשורת מורה-תלמיד בדומה למערכת משו"ב שבה בית הספר שלי משתמש. דרך התוכנה יכולים המורה והתלמיד לתקשר באמצעות צ'אט שכתבתי, המורה יכול להעלות ציונים לתלמיד והתלמיד יוכל לראות את כל הציונים שלו, לחשב ממוצעים וכו'. כמו כן בתוך התוכנה יהיה לוח מבחנים שבו המורה יוכל לקבוע תאריך ויום בו ירצה לקיים מבחן לתלמידים והתלמידים יוכלו להסתכל בלוח המבחנים ולראות באיזה יום יתקיים איזה מבחן. יתרה מזאת, בתוך התוכנה תהיה אפשרות להעלות קבצים כמו עבודות ושיעורים ולוח הודעות שדרכו יוכלו המורה והתלמיד להתכתב מבלי שיהיו מחוברים בו זמנית, או כמו שזה נקרא בשפה המקצועית תקשורת א-סינכרונית.

לפני התחלת כתיבת הפרויקט ביקשתי מן המורה המנחה שלי שיגיד לי אפשרויות ופונקציות הקיימות במערכת משו"ב. המורה נתן לי מספר רעיונות לפונקציות שאכתוב בתוך התוכנה. לאחר מכן התחלתי לחשוב בעצמי מה מפריע לי בתוכנה הקיימת, הדבר העיקרי שהפריע לו הוא שכאשר אני שולח הודעה למורה יכולים לקחת שעות ואף ימים עד שהוא מחזיר לי תשובה ולכן החלטתי לפתח צ'אט אונליין בו יוכלו המורה והתלמיד לדבר יחדיו ולקבל תשובות מיידיות. בנוסף לכך מצד של התלמיד הוספתי פונקציות המוכרות לי כתלמיד המשתמש באפליקציית משו"ב.

הבעיה העיקרית שבה נתקלתי היא בכתיבת הצ'אט מכיוון שבעקבות העובדה שרציתי שהמורה והתלמיד יוכלו לדבר ביחד וגם שהצ'אט ירוץ במקביל לתוכנה נתקלתי במצב שבו ברגע שאני מפעיל את התוכנה ואת השרת של הצ'אט נוצר מצב שרצות 2 לולאת אין סופיות במקביל ולכן במידה ואני רוצה לצאת מהתוכנית לפני שניתקתי את הקשר עם השרת התוכנה הייתה נתקעת ולאחר מכן קורסת, אך אם הייתי מתחבר לצ'אט ומנתק את הקשר עם השרת התוכנית הייתה רצה כרגיל.

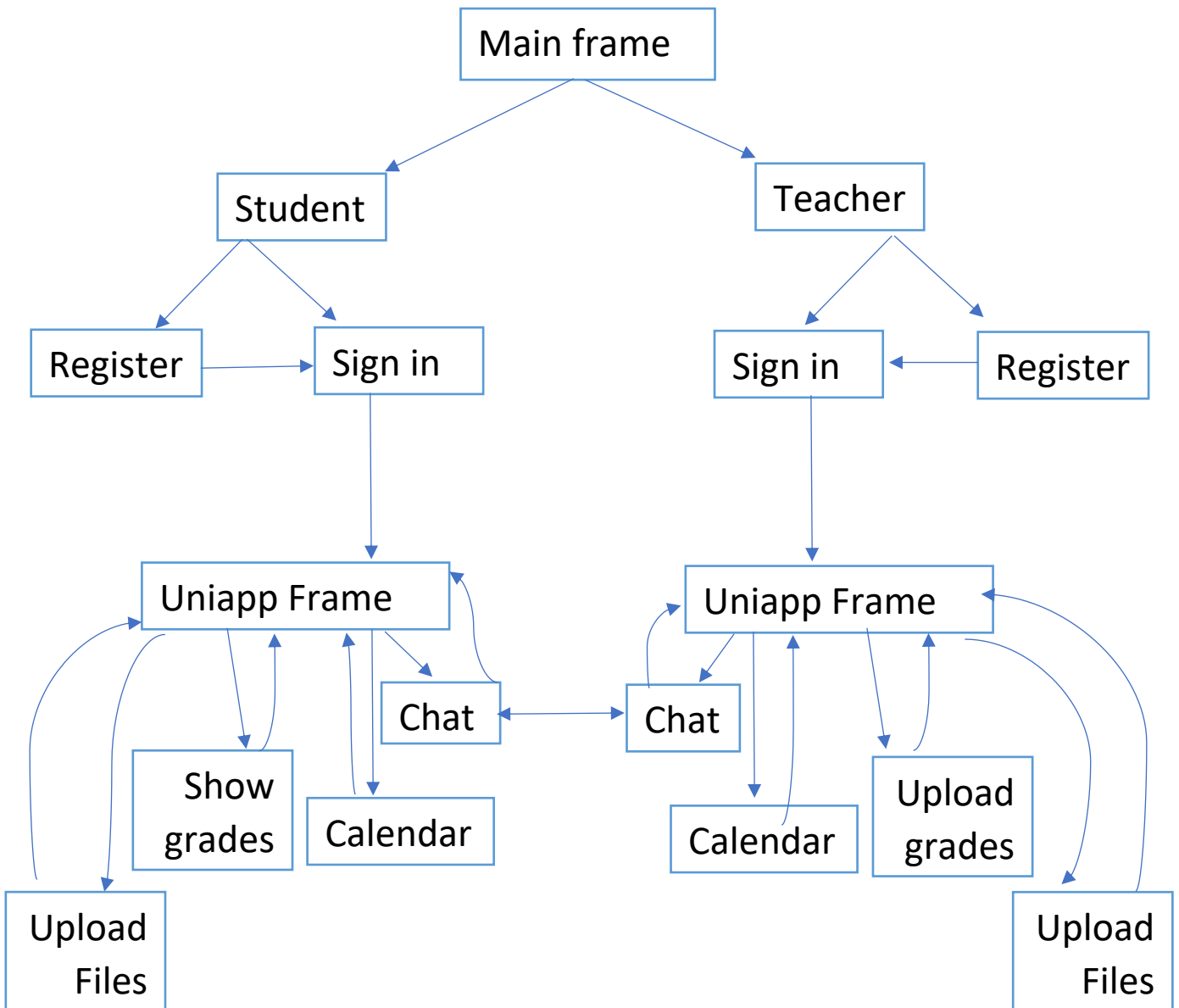
בחרתי את הנושא מכיוון שלפני מספר שנים בניתי תוכנה דומה לתוכנה זו, אך בשפת ג'אבה, במסגרת קורס חיצוני שהשתתפתי בו, אך כיוון שבאותו הזמן לא ידעתי רשתות לא יכולתי לקשר בין המערכת של המורה למערכת של התלמיד. כאשר המורה המלווה לימד אותנו

רשתות ידעתי באותו הרגע שעכשיו זו ההזדמנות שלי להשלים את המלאכה שהתחלתי מספר שנים קודם.

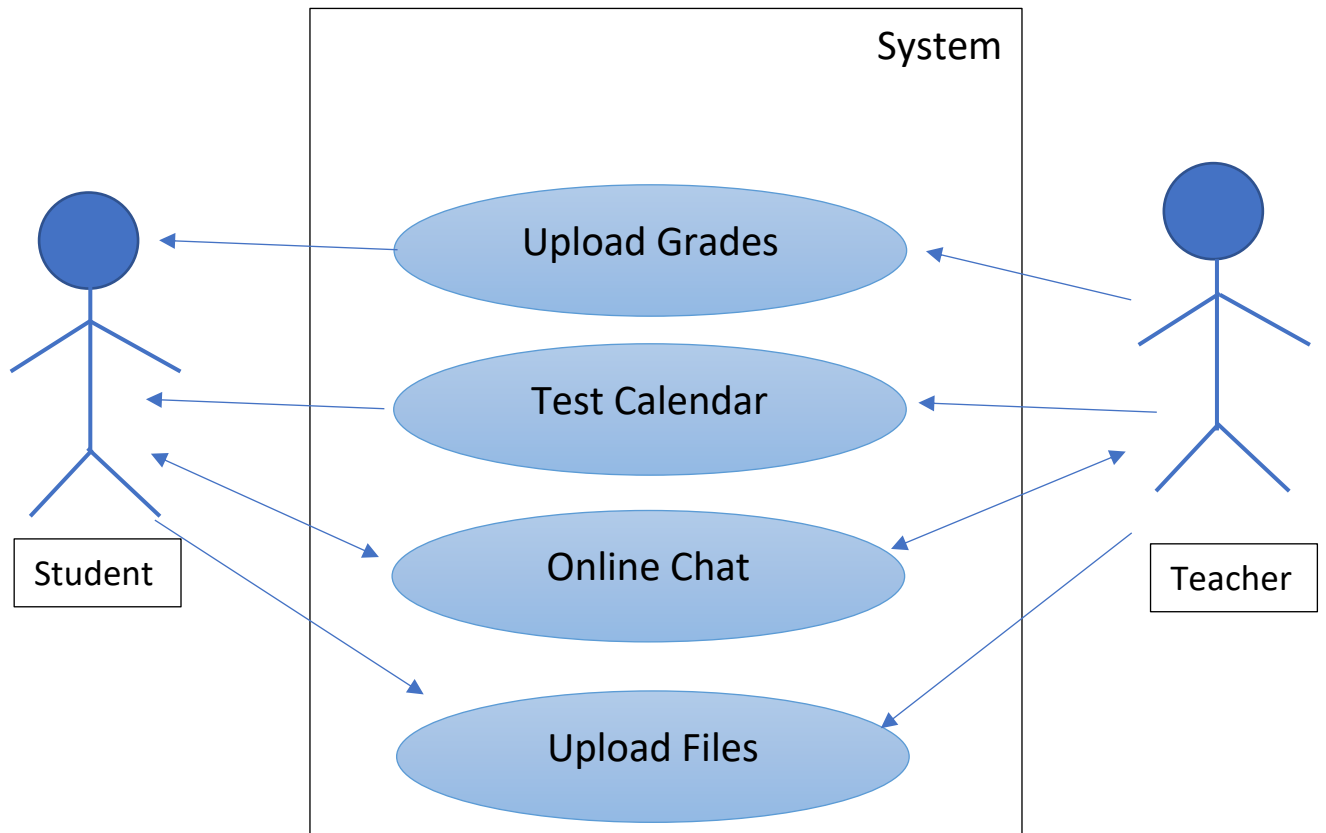
המוטיבציה שלי לעבודה הייתה האכזבה שנחלתי מספר שנים קודם מכך שהפרויקט שבניתי לא היה מושלם, ולא עבד כמו שרציתי, ולכן כל הזמן עמדה בראשי המחשבה שהפעם הוא יהיה מושלם ובדיוק כמו שחלמתי עליו.

מבנה הפרויקט:

תרשים Top-Down level Design:



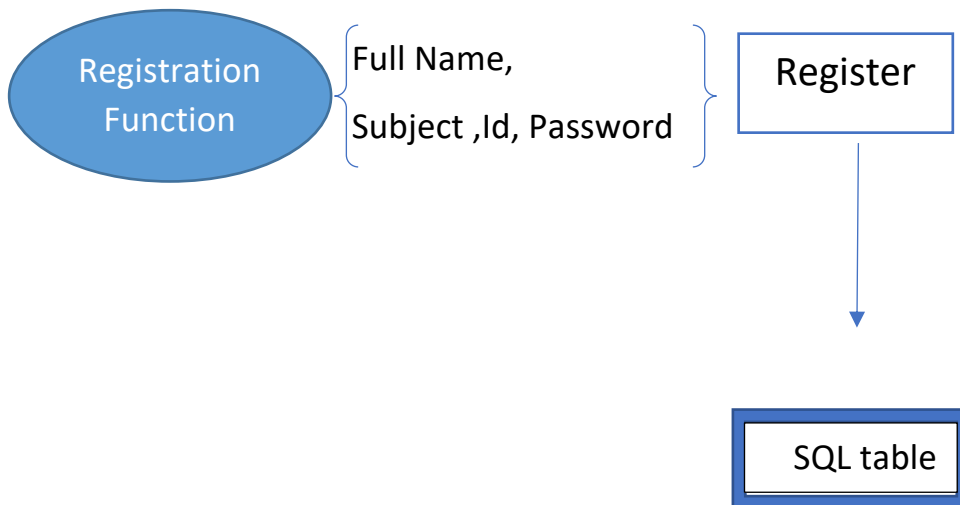
תרשים Use Case המתאר את מבנה הפרויקט:



תיאור הפונקציות העיקריות בפרוייקט:פונקציית Register:

מטרת פונקצייה זו היא לקבל מן המשתמש את הנתונים הנדרשים ולקבל ממנו מידע על תפקידו בבית הספר. לדוגמא: אם המשתמש הינו מורה המערכת מבקשת ממנו תחילה את שמו הפרטי ושם המשפחה שלו, לאחר מכן את המקצוע בו הוא מלמד, ולבסוף את תעודת הזהות שלו שתהווה שם המשתמש לתוכנה ואת הסיסמא עימה הוא יוכל להיכנס.

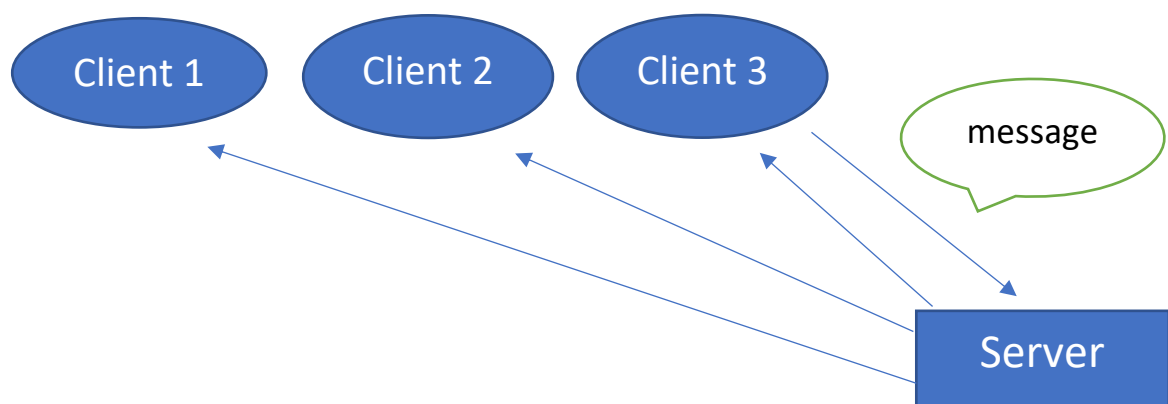
לאחר מכן בלחיצה על הכפתור Register המערכת לוקחת את כל המידע שהמשתמש נתן ומאחסנת אותו בטבלת sql ספציפית בהתאם לתפקידו(מורה/תלמיד) לאחר שהמשתמש הזין את הנתונים אין צורך שיזין אותם כל פעם מחדש אלא הוא יוכל להיכנס ללא הפרעה. ישנם 2 טבלאות שונות המאחסנות את נתוני המשתמשים. טבלה אחת הינה טבלה שמכילה את הפרטים של המורים וטבלה נוספת שמכילה את הפרטים של התלמידים. מצורפת דוגמא הממחישה את תהליך ההרשמה של מורה לאפליקצייה:



כפי שניתן לראות בתיאור המשתמש מזין את הנתונים בעצמו כגון: שם מלא מקצוע ת.ז. וסיסמא, ובלחיצה על כפתור ההרשמה הנתונים נכנסים לתוך טבלת SQL לפי העמודות המתאימות.

פונקציית Online Chat:

באמצעות פונקציה זו המשתמשים יכולים לדבר אחד עם השני. הפונקציה פועלת בשיטת שרת-לקוח, כאשר אל השרת יכולים להתחבר מספר לקוחות במקביל ולדבר בו זמנית. שיטה זו פועלת על פי העיקרון שכאשר לקוח רוצה לשלוח הודעה, הוא כותב אותה ובלחיצת כפתור מעביר אותה אל השרת, והשרת משדר אותה לכל שאר המשתמשים המחוברים. השידור מתבצע באמצעות הדפסת ההודעה הנשלחת על גבי מסך Text שנמצא על gui של הצ'אט. לפני התחלת השיחה על הלקוח להתחבר אל השרת. החיבור מתבצע כאשר המשתמש מקליד את שמו, השרת מזהה זאת, ולאחר הזיהוי הוא מחבר את הלקוח לשיחה ומעדכן את שאר הנוכחים בשיחה שהצטרף אדם חדש לשיחה.



התרשים המצורף מתאר את הקשרים בין הלקוחות לשרת. במקרה זו בחרתי שיהיו שלושה לקוחות מחוברים בו זמנית לשיחה: Client 1, Client 2, Client 3. התרשים מראה שלקוח 3 רוצה לשלוח הודעה, לשם כך הוא נדרש לשלוח את ההודעה תחילה אל השרת. השרת מקבל את ההודעה מלקוח 3, מעבד אותה ולאחר מכן הוא משדר אותה אל כל הלקוחות המחוברים לשיחה. כמו כן, לצ'אט ישנה מטרה נוספת והיא, לעזור למורה לנהל שיעור וירטואלי, כפי שאנו ניהלנו בעקבות משבר הקורונה שפקד את מדינת ישראל.

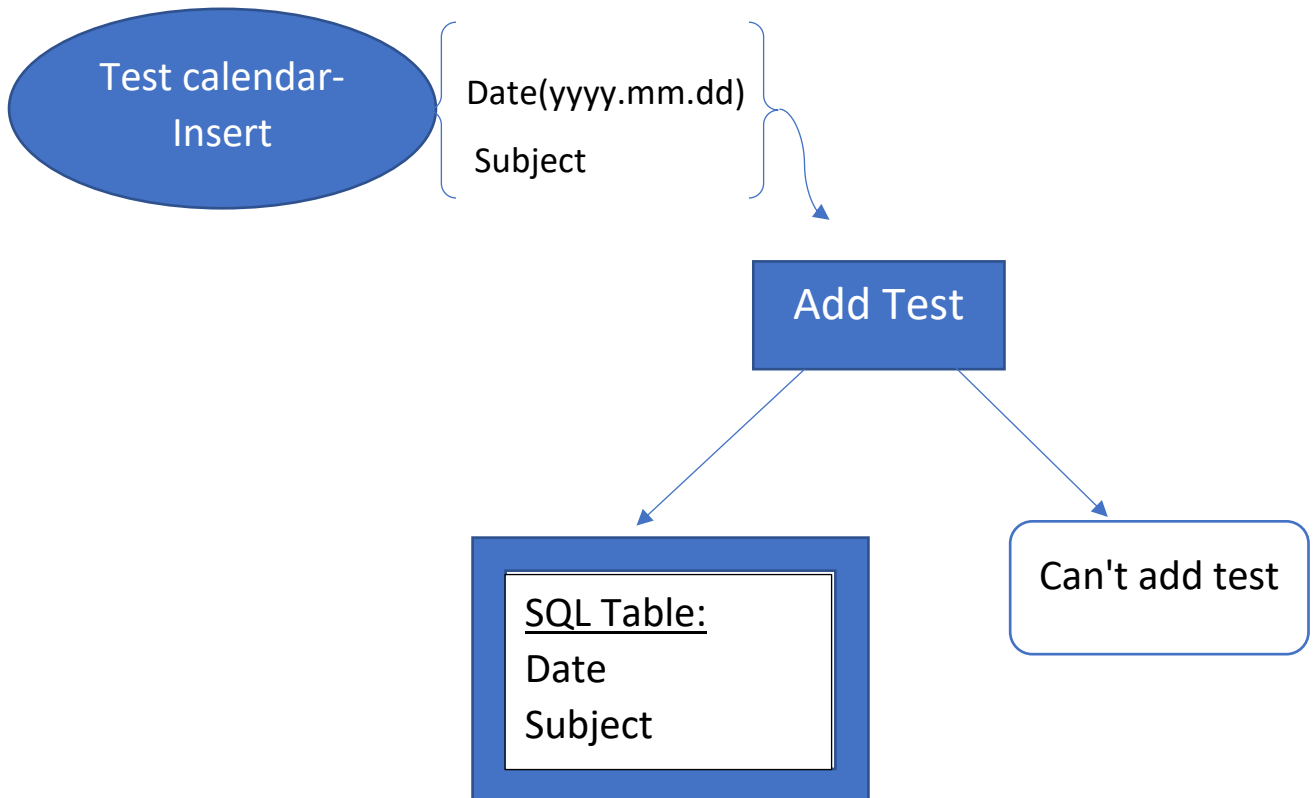
פונקציית Calendar:

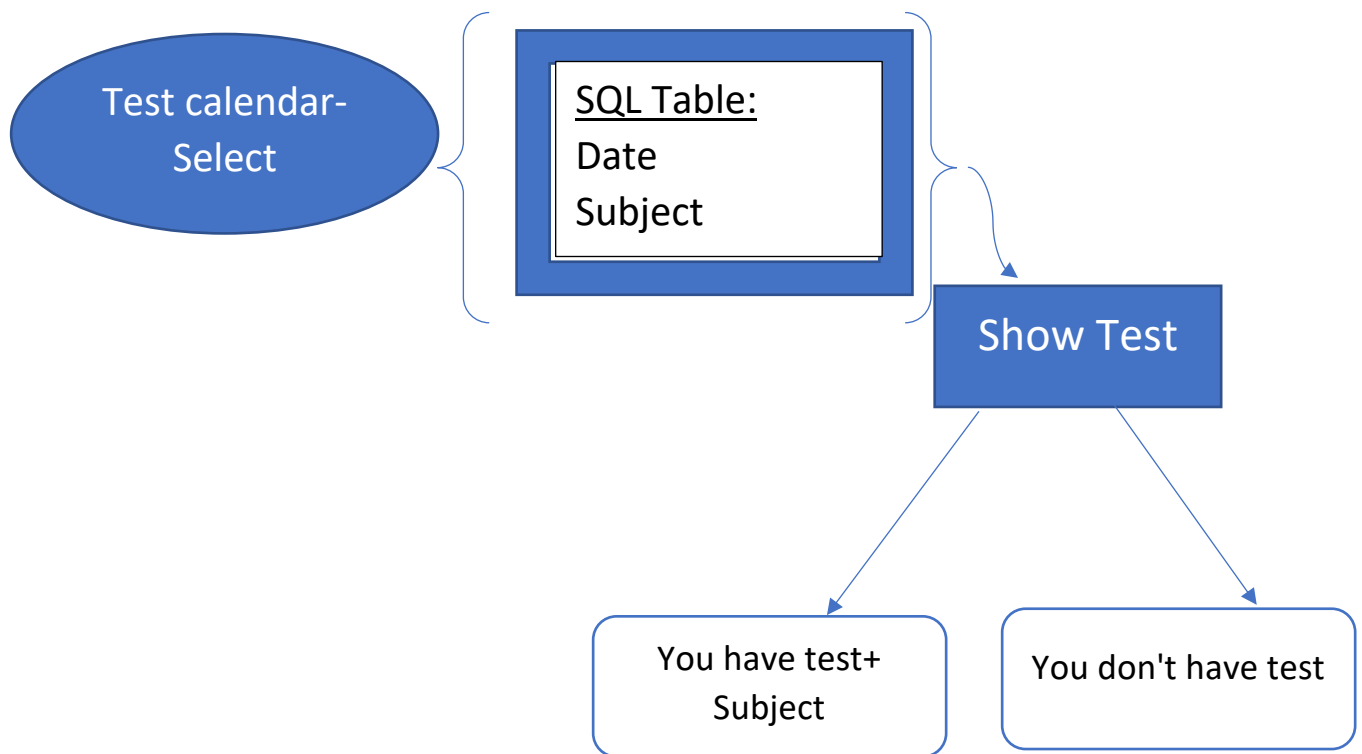
מטרת פונקציה זו היא שהמורה בוחר תאריך בו הוא רוצה לקיים מבחן עבור התלמידים. לאחר שהמורה מסמן תאריך בו הוא מעוניין לקיים את המבחן הוא לוחץ על כפתור Add Test. לאחר לחיצת הכפתור הפונקציה מקבלת את התאריך והמקצוע שבהם מתקיים מבחן, היא עוברת על ה-database של המבחנים ובודקת האם בתאריך המצוין יש מבחן אחר, במידה ויש היא שולחת הודעה למשתמש שאין אפשרות לקיים את המבחן והוא צריך

לבחור תאריך אחר. במידה ואין מבחן בתאריך זה היא שומרת בטבלת sql את התאריך וסוג המקצוע שמתקיים בו מבחן.

כאשר תלמיד רוצה לבדוק האם בתאריך מסוים יש מבחן הוא יילך אל לוח המבחנים, יילחץ על התאריך שבו הוא רוצה לבדוק אם יש מבחן, ולאחר מכן יילחץ על הכפתור Show Test. לאחר הלחיצה על הכפתור הפונקציה תבצע עבודה הפוכה ממה שעשתה קודם, היא תלך אל ה-database ותבדוק אם בתאריך המבוקש יש מבחן. במידה ואין מבחן היא תשלח הודעת שגיאה, ובמידה ויש מבחן היא תשלח את סוג המקצוע שבו מתקיים המבחן.

תרשים הכנסת מבחן אל לוח המבחנים:



תרשים בדיקה האם יש מבחן בתאריך מסוים:פונקציית Grades Upload:

מטרתה של פונקציה זו היא שהמורה יוכל להעלות לתלמיד מסוים ציון במקצוע ספציפי, ואילו התלמיד יוכל לבחור באיזה מקצוע הוא רוצה לראות את הציונים שלו. בנוסף לתלמיד ניתנה אפשרות לצפות בכל גיליון הציונים שלו. הפעולה מתבצעת כך:

תחילה כאשר המורה רוצה להעלות ציונים, נפתחת בפניו רשימה של כל התלמידים הרשומים במערכת- למעשה זה כל התלמידים הנמצאים בטבלת ה-SQL. לאחר מכן המורה בוחר את המקצוע בו הוא מלמד ונותן את הציון.

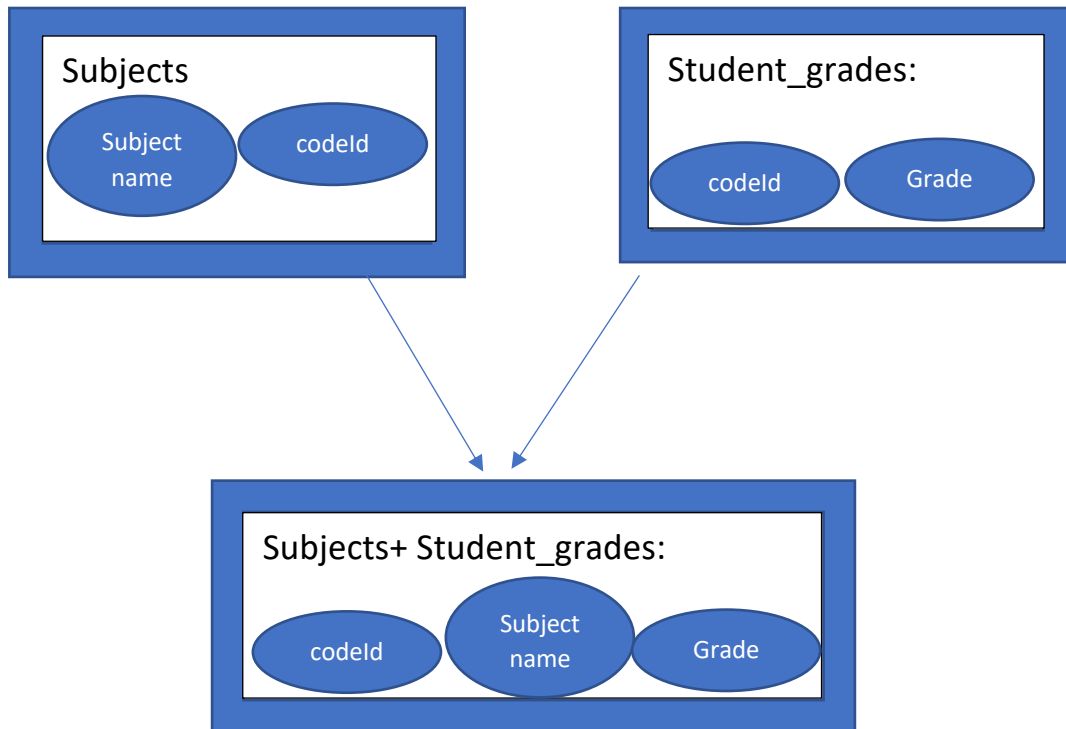
כאשר המורה לוחץ על הפתור העלאת הציון, הציון הנבחר נכנס לטבלת SQL ונשמר שם ביחד עם קוד המקצוע הנבחר.

כאשר תלמיד רוצה לראות את הציון במקצוע מסוים קורה הדבר הבא:

הפעולה לוקחת את שם המקצוע שהתלמיד בחר, מוצאת בטבלת המקצועות את קוד המקצוע. לאחר מכן נוצר קשר בין 2 טבלאות SQL שכתבתי. הקשר קורה באמצעות מילת מפתח codeld. כתוצאה מכך נוצרת טבלה גדולה המכילה את כל הפרטים של התלמיד ביניהם: שם התלמיד, שם המקצוע והציון. הטבלה מציגה לכל תלמיד את הציונים שלו לפי

תעודת הזהות שהוא מכניס.

בתיאור הבא ניתן לראות את הקשר בין 2 הטבלאות הנ"ל:



מדריך למשתמש:

לצורך כתיבת הפרויקט שלי השתמשתי בתוכנת PyCharm

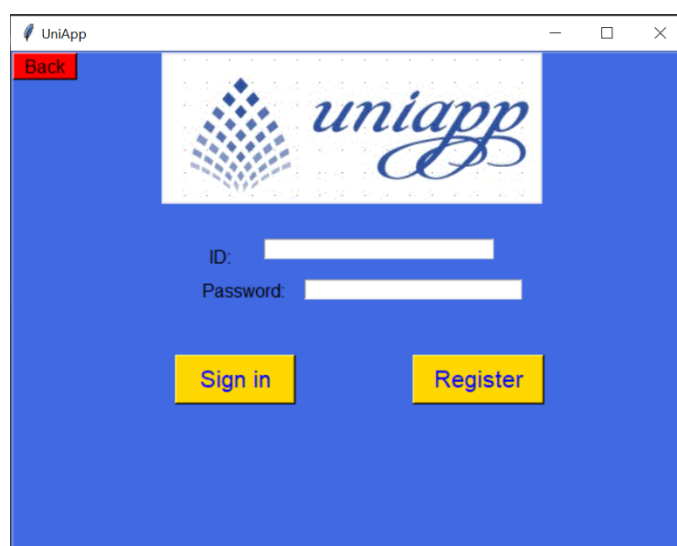
קבעתי בהגדרות התוכנה את ה-`interpreter` של התוכנית כ-`python 3.6`.

כעת ניתן להריץ את התוכנית שלי. לאחר פתיחת המסך תוצג בפניך השאלה האם אתה מורה או תלמיד ותוכל לענות על כך לפני בחירתך.



לאחר בחירה של מורה או תלמיד יוצג בפניך מסך הכניסה שם שתצטרך להזין את תעודת הזהות שלך ואת הסיסמא שלך.

שים לב: במידה ואינך רשום במערכת תצטרך ללחוץ תחילה על כפתור ה-`Register`.



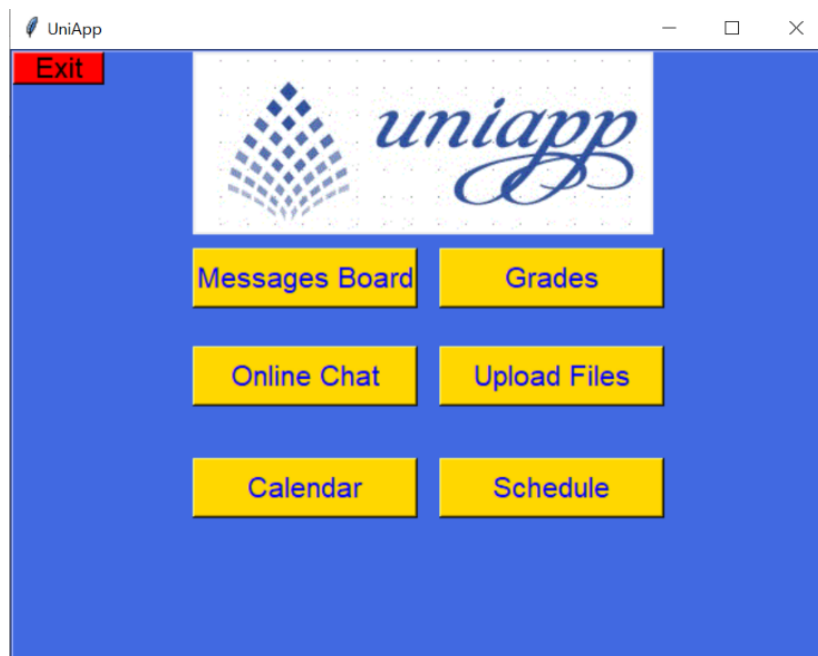
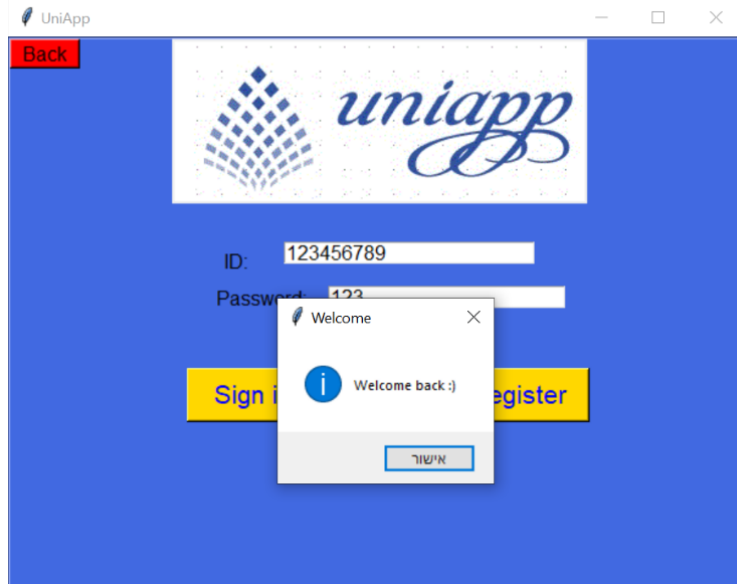
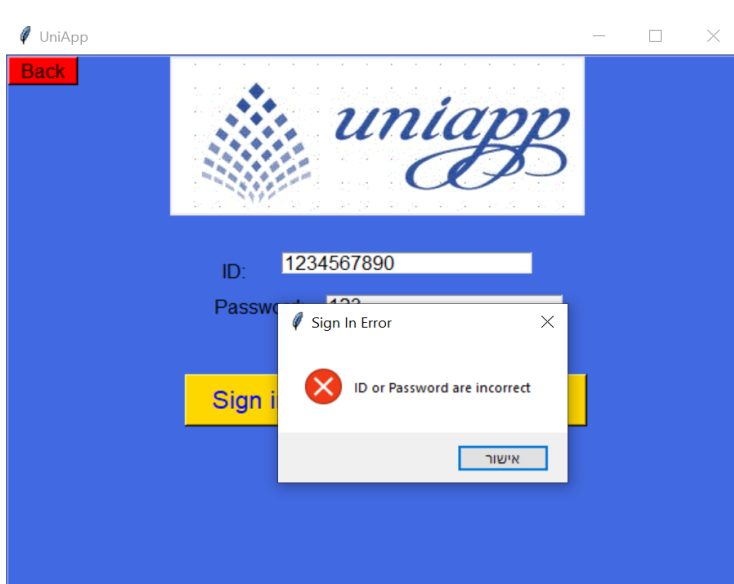
במידה ובחרת מורה יוצג בפניך מסך ההרשמה הבא:

The screenshot shows a web browser window titled 'UniApp'. In the top left corner, there is a red 'Back' button. The main content area has a blue background with the 'uniapp' logo at the top. Below the logo, there are input fields for 'First Name:' and 'Last Name:'. The 'Subject:' field has a dropdown menu open with the following options: Math, Physics, English, Chemistry, Bible, Citizenship, Sport, History, Computers, and Literature. Below these fields are 'ID:' and 'Password' fields. At the bottom, there are two yellow buttons: 'Clear' and 'Register'.

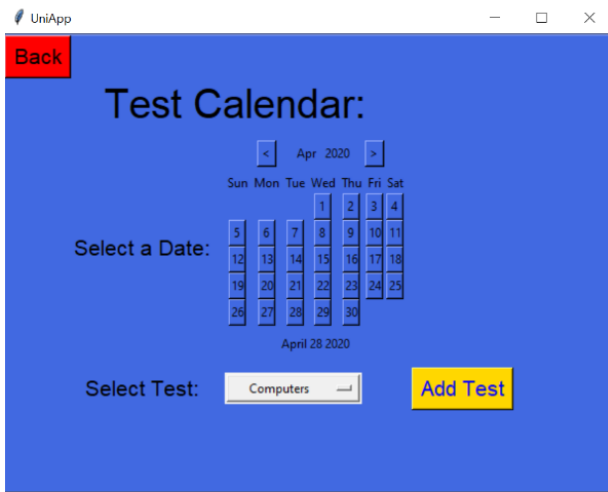
במידה ובחרת תלמיד יוצג בפניך המסך הבא:

The screenshot shows a web browser window titled 'UniApp'. In the top left corner, there is a red 'Back' button. The main content area has a blue background with the 'uniapp' logo at the top. Below the logo, there are input fields for 'First Name:' and 'Last Name:'. The 'Class:' field has a dropdown menu open with the following options: 6th class, 7th class, 8th class, 9th class, 10th class, 11th class, and 12th class. Below these fields are 'ID:' and 'Password' fields. At the bottom, there are two yellow buttons: 'Clear' and 'Register'.

לאחר ההרשמה תוכל לחזור אחורה בעזרת כפתור ה-Back ולהיכנס לתוכנה.
שים לב: תעודת הזהות שלך חייבת להכיל 9 ספרות בלבד.
במידה ושם המשתמש והסיסמא נכונים המערכת תשלח לך הודעה שאתה רשאי להיכנס,
ואם לא הוא יפרסם הודעת שגיאה.

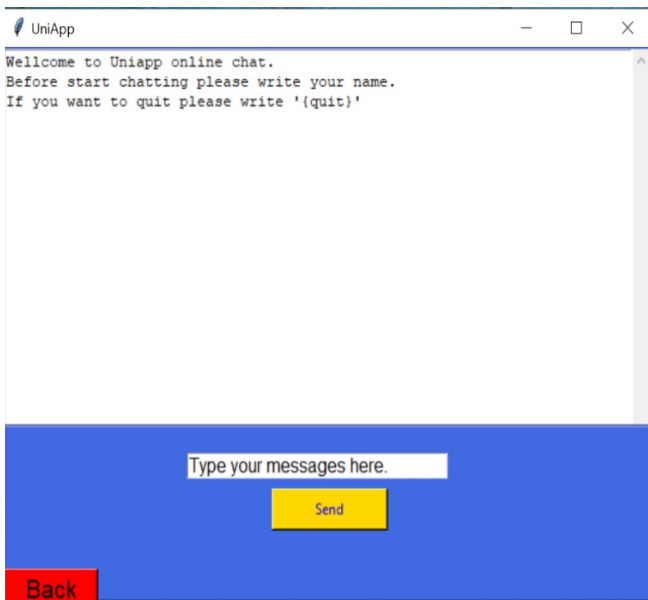


כעת נכנסת אל תוך האפליקציה ותוכל לבחור מה תרצה לעשות. במידה ובחרת מורה תוכל לבחור להעלות ציונים בלחיצה על כפתור Grades, להיכנס אל הצ'אט בלחיצה על הכפתור Online Chat, לקבוע תאריך למבחן על ידי לחיצה על הכפתור Calendar.



לוח מבחנים

העלאת ציונים לתלמיד



צ'אט אונליין



העלאת קבצים

בסיס הנתונים:

בפרויקט שלי בסיס הנתונים הוא החלק המרכזי ביותר. כל הנתונים שהמשתמשים מזינים נשמרים בטבלאות SQL ייחודיות שכתבתי למספר מטרות שאפרט עליהם מיד. חשוב לציין שכל העמודות בטבלאות הן הכרחיות ואסור להשאיר אף עמודה ריקה. במידה והמשתמש שוכח להוסיף פרט לעמודה התוכנה תשלח הודעת שגיאה שחסר נתון מסוים

טבלה ראשונה- students:

טבלה זו מכילה את כל הפרטים של התלמיד הנרשם לאפליקציה. העמודות שהטבלה מכילה הינם:

Student_name-Text	Class-Text	ID-Integer	Password-Text
-------------------	------------	------------	---------------

Student name- בעמודה זו נשמר השם המלא של התלמיד: שם פרטי + שם משפחה. סוג העמודה יהיה טקסט מכיוון שהשם הפרטי ושם המשפחה מכילים אותיות בלבד.

Class- בעמודה זו נשמרת הכיתה בה התלמיד לומד. סוג העמודה יהיה טקסט מכיוון שה- option menu לפיו בוחר התלמיד את הכיתה, הוא מסוג טקסט.

ID- בעמודה זו נשמרת תעודת הזהות של התלמיד שמשתמש עבורו את שם המשתמש איתו הוא נכנס לאפליקציה. לעמודה זו יהיה שימוש נוסף בהמשך עבור טבלה אחרת. סוג העמודה יהיה מספרים שלמים מכיוון שהיא יכולה לקבל רק ספרות בין 0-9.

Password- בעמודה זו נשמרת הסיסמא של המשתמש לאפליקציה. סוג העמודה יהיה טקסט מכיוון שהסיסמא תוכל להכיל בתוכה גם ספרות וגם אותיות.

טבלה שנייה - teachers:

טבלה זו מכילה את כל הפרטים של המורים הנרשמים לאפליקציה. העמודות שהטבלה מכילה הינם:

FirstN-Text	LastN-Text	Subject-Text	ID-Integer	Password-Text
-------------	------------	--------------	------------	---------------

FirstN- בעמודה זו נשמרים השמות הפרטיים של המורים. סוג העמודה יהיה טקסט מכיוון שעמודה זו תכיל אותיות בלבד.

LastN- בעמודה זו נשמרים שמות המשפחה של המורים. סוג העמודה יהיה טקסט מכיוון שעמודה זו תכיל אותיות בלבד.

Subject- בעמודה זו נשמר המקצוע שהמורה הנרשם מלמד. סוג העמודה יהיה טקסט מכיוון שה-option menu לפיו בוחר המורה את המקצוע, הוא מסוג טקסט.

ID- בעמודה זו נשמרת תעודת הזהות של המורה שמשמשת עבורו את שם המשתמש איתו הוא נכנס לאפליקציה. סוג העמודה יהיה מספרים שלמים מכיוון שהיא יכולה לקבל רק ספרות בין 0-9.

Password- בעמודה זו נשמרת הסיסמא של המשתמש לאפליקציה. סוג העמודה יהיה טקסט מכיוון שהסיסמא תוכל להכיל בתוכה גם ספרות וגם אותיות.

טבלה שלישית - tests:

טבלה זו אחראית על שמירת הנתונים של לוח המבחנים, אליה מזינים המורים את תאריך המבחן והמקצוע של המבחן, והתלמידים לוקחים ממנה פרטים על לוח המבחנים. העמודות שהטבלה מכילה הינם:

Yyyymmdd-Text	Subject-Text
---------------	--------------

Subject- עמודה זו מכילה את שם המקצוע שבו ייערך המבחן. סוג העמודה יהיה טקסט מכיוון שה-option menu לפיו בוחר המורה את המקצוע שבו יהיה מבחן, הוא מסוג טקסט.

Yyyymmdd- עמודה זו מכילה את התאריך שבו יתקיים המבחן. משמעות השם היא התאריך שבו המבחן יתקיים כאשר yyyy מסמל את השנה, mm את החודש, dd את היום. המטרה של כך היא שבמידה והמורה ירצה לבצע מיון של המבחנים יהיה לו פשוט יותר.

טבלה רביעית- subjects:

טבלה זו משמשת לאחסון קוד מקצוע לימוד, ושם מקצוע הלימוד. אני משתמש בטבלה זו על מנת לבצע את תהליך הזנת הציונים לתלמיד, למעשה אני מחבר את טבלה זו עם הטבלה החמישית עליה אפרט בהמשך וכך מתבצע תהליך העלאת הציונים. העמודות שהטבלה מכילה הינם:

Codeld-Integer	SubName-Text
----------------	--------------

SubName- בעמודה זו נשמר השם של המקצוע בו ניתן הציון. סוג העמודה הוא טקסט מכיוון ששם המקצוע מכיל אותיות בלבד.

Codeld- בעמודה זו נשמר קוד המקצוע של המקצוע הספציפי. חשוב לציין שקוד המקצוע הוא קבוע ולא ניתן לשינוי. סוג העמודה הוא מספרים מכיוון שהקודים הם מספרים שלמים שלא ניתנים לשינוי. הקודים של המקצועות הינם:

- 1- מתמטיקה
- 2- פיזיקה
- 3- מתמטיקה
- 4- כימיה
- 5- תנ"ך
- 6- אזרחות
- 7- ספורט
- 8- היסטוריה
- 9- מחשבים
- 10- ספרות
- 11- ערבית

טבלה חמישית- student grades:

בטבלה זו נשמרים עוד נתונים העוזרים לתהליך העלאת הציונים. למעשה בקוד שכתבתי ישנה פעולה המחברת בין טבלה זו לטבלה הרביעית אשר נקראת פעולת JOIN, למעשה פעולה זו "מחברת" את שתי הטבלאות ויוצרת טבלה גדולה המכילה את שתי הטבלאות לפי מילת מפתח שמקרה זה נקראת codeID. העמודות שהטבלה מכילה הינם:

ID-Integer	CodeID-Integer	Grade-Integer
------------	----------------	---------------

Grade- עמודה זו מכילה את הציון שהמורה נותן לתלמיד על המבחן שביצע. סוג העמודה הוא מספרים שלמים מכיוון שלרוב הציונים בבית הספר הם מספרים שלמים ולכן גם אני החלטתי לעשות כך.

Codeid- בעמודה זו נשמר קוד המקצוע של המקצוע הספציפי. חשוב לציין שקוד המקצוע הוא קבוע ולא ניתן לשינוי. סוג העמודה הוא מספרים מכיוון שהקודים הם מספרים שלמים שלא ניתנים לשינוי. הקודים של המקצועות הם אותם הקודים כפי שפירטתי קודם.

ID- בעמודה זו נשמרת תעודת הזהות של התלמיד שמשמשת במקרה זה לצורך התאמת הציון לתלמיד מסוים. סוג העמודה יהיה מספרים שלמים מכיוון שהיא יכולה לקבל רק ספרות בין 0-9.

טבלה שישית- files_name:

בטבלה זו נשמרים שמות הקבצים שהמשתמשים מעלים לשרת. בטבלה זו יישמרו השמות בלבד וכאשר התלמיד או המורה מחליטים להעלות את הקבצים, הקובץ עצמו נשמר בשרת ייחודי, ושם הקובץ וסוגו יישמרו בטבלת SQL. העמודות שהטבלה מכילה הינם:

מדריך למפתח:

בכתיבת הפרויקט החלטתי לפצל את הקוד המלא לקטעי קוד קטנים יותר הנמצאים בתוך תיקיה שפתחתי הנקראת utils והיא נמצאת בתוך תיקיית הפרויקט בתור python package. כל קטע קוד הנמצא בתוך התיקיה מכיל פקודות המיועד לשימוש מסוים, כך שנוצר סדר בקוד והוא מחולק בצורה הגיונית. הקובץ העיקרי של הפרויקט נקרא app, והוא מכיל בעיקרו את הממשק הגרפי של הפרויקט. את החיבור בין הקוד המרכזי לקטעי הקוד הקטנים ביצעתי באמצעות פקודת import, וכך יכולתי להשתמש בפונקציות שכתבתי.

:App

זוהי המחלקה הראשית של הפרויקט. היא נמצאת בתוך המחלקה הראשית project2. אל מחלקה זו מתבצעים כל ה import של המחלקות האחרות ע"י שימוש בפקודה זו:

```
from utils.frames import *
```

זאת דוגמה לקריאה למחלקת frames שאחראית על החלפת המסכים באפליקציה.

במחלקה זו נמצאים כל העיצובים של המסכים כפי שבחרתי לעצב. העיצוב מתבטא בשורות קוד כפי שניתן לראות:

```
frame1 = Frame(root, relief=GROOVE, bg="royal blue", borderwidth="2", width=605)
frame1.place(relx=0.0, rely=0.0, relheight=1.01, relwidth=1.01)

lbl = Label(frame1, image=img)
lbl.config(image=img)
lbl.image = img
lbl.pack()

stB = Button(frame1, text="Student", bg="gold", fg="blue", font=("Ariel", 16))
stB.place(relx=0.18, rely=0.55, height=54, width=107)
stB.bind("<Button-1>", lambda event: show_frame(logFrame, event))

teachB = Button(frame1, text="Teacher", bg="gold", fg="blue", font=("Ariel", 16))
teachB.place(relx=0.63, rely=0.55, height=54, width=107)
teachB.bind("<Button-1>", lambda event: show_frame(tlogFrame, event))

Quit = Button(frame1, text="Quit", bg="red", fg="black", font=("Ariel", 16))
Quit.place(relx=0.02, rely=0.04, height=44, width=107)
Quit.bind("<Button-1>", exitt)

hellbl = Label(frame1, text="Please choose if you are Student or Teacher", bg="royal blue",
fg="black",
font=("Ariel", 16))
hellbl.place(relx=0.14, rely=0.42, height=31, width=450)
```

אלו שורות הקוד שיצרו את המסך הבא:



מלבד עיצוב המסכים במסמך זה נמצאות מספר פעולות הכרחיות שחייבות להימצא רק במסמך זה לדוגמא:

```
def all_grades_temp(gradebox,event):
    all_grades_list=all_grades(event)
    for i in all_grades_list:
        gradebox.insert(END,i)
```

הפונקציה `all_grades_temp` אחראית להדפסת כל הציונים על המסך, והיא חייבת להימצא כאן מכיוון שרק כאן מוגדר ה-`listbox` שעליו מודפסים הציונים.

:Frames

מחלקה זו נמצאת בתוך ספריית utils שיצרתי בפרויקט. המטרה של המחלקה היא להחליף בין המסכים השונים בפרויקט, לנקות את המסך מהתוכן שהשתמש כתב. מטרה נוספת היא הגדרה של תנאים שחייבים להתקיים. למשל:

```
def testVal1(inStr, acttyp):
    if acttyp == '1': #insert
        if not inStr.isdigit():
            return False
        return True
```

הפונקציה testVal1 מכילה בתוכה תנאי שמחייב שתעודת הזהות של המשתמש תהיה ספרות בלבד ולא תכיל בתוכה אותיות. כמו כן :

```
def chat_frame():
    """ Activated by pressing action2 button. """
    th1 = Thread(target=server_main)
    th1.start()
    mainclient2()
```

פונקציית chat_frame אחראית על 2 דברים חשובים. תחילה היא אחראית על הפעלת הסרבר כך שהוא לא ירוץ מתחילת התוכנית אלא יתחיל לפעול רק כאשר לוחצים על הכפתור המתאים. כמו כן, לאחר הפעלת הסרבר הפונקציה קוראת למחלקת קליינט וכך מתחיל הצ'אט.

:Create sql

מחלקה זו נמצאת בתוך ספריית project2 שיצרתי בפרויקט. המטרה של המחלקה היא ליצור את ה-database שישמור את כל הנתונים בתוכנית וידאג שהם לא נמחקים כאשר סוגרים את התוכנה. את מחלקה זו מריצים פעם אחת בלבד לפני ההתחלה הראשונית של הפרויקט ולאחר מכן אין צורך להפעיל שוב. על מנת לעשות זאת היה צריך תחילה לקשר את המחלקה עם database מסוים כפי שניתן לראות כאן:

```
conn = sqlite3.connect('example.db')
cursor = conn.cursor()
```

לאחר שעשיתי זאת בחרתי לוודא שהטבלה אכן ריקה ולא מכילה קבצים מיותרים שיכולים להשפיע על הפרויקט ולגרום לבעיות מסוגים שונים. עשיתי זאת באמצעות פקודת DROP, שאחראית למחיקת כל הנתונים:

```
cursor.execute("""DROP TABLE students""")
```

לבסוף לצורך יצירת הטבלה השתמשתי בפקודה הבאה:

```
cursor.execute(""" CREATE TABLE IF NOT EXISTS students (student_name TEXT, class TEXT, id integer, password TEXT);""")
```

למעשה לאחר כתיבת שורה זו נוצרת הטבלה ב-database עם העמודות: שם תלמיד, כיתה, ת.ז, סיסמא.

:Sqltable

מחלקה זו נמצאת בתוך ספריית utils שיצרתי בפרויקט. המטרה של המחלקה היא לנהל את כל הפעולות הקשורות לטבלאות ה-sql. למשל, בתוך המחלקה ישנם פעולות הקשורות להרשמה של המשתמשים לאפליקציה, כפי שניתן לראות:

```
def user_insert(table_name,firstN,lastN,subject,id,password,event):

    tablen=table_name
    firstN = firstN
    lastN = lastN
    subject =subject
    id = id
    password = password

    if tablen=='students':
        if len(firstN) == 0 or len(lastN) == 0 or len(id) == 0 or len(subject) == 0 or len(id) == 0 or len(
            password) == 0:
            messagebox.showerror("Error", "something missing")
        elif len(id) != 9:
            messagebox.showerror("Error", "Illegal id")
        else:
            cursor.execute("SELECT * FROM teachers WHERE id = ?", (id,))
            data1 = cursor.fetchall()
            cursor.execute("SELECT * FROM students WHERE id = ?", (id,))
            data = cursor.fetchall()
            full_name=firstN+" "+lastN
            if len(data) == 0 and len(data1) == 0:
                cursor.execute("INSERT INTO students (student_name,class,id,password)
VALUES(?,?,?,?)"
                    , (full_name, subject, id, password))
                messagebox.showinfo("Succeed", "Registration is succeeded")
            else:
                messagebox.showerror("Error", "You are already signed up")
            conn.commit()

    else:
        if len(firstN) == 0 or len(lastN) == 0 or len(id) == 0 or len(subject) == 0 or len(id) == 0 or len(
            password) == 0:
            messagebox.showerror("Error", "something missing")
        elif len(id) != 9:
            messagebox.showerror("Error", "Illegal id")
        else:
            cursor.execute("SELECT * FROM teachers WHERE id = ?", (id,))
            data1 = cursor.fetchall()
            cursor.execute("SELECT * FROM students WHERE id = ?", (id,))
            data = cursor.fetchall()
            if len(data) == 0 and len(data1) == 0:
                cursor.execute("INSERT INTO teachers (firstN,lastN,subject,id,password)
VALUES(?,?,?,?,?)"
                    , (firstN, lastN, subject, id, password))
                messagebox.showinfo("Succeed", "Registration is succeeded")
            else:
                messagebox.showerror("Error", "You are already signed up")
```



```
conn.commit()
print('nothing')
```

כפי שניתן לראות הפעולה user_insert מקבלת את כל הנתונים שמזין המשתמש ביניהם, השם שלו, כיתה/מקצוע לימוד, ת.ז וסיסמא. כמו כן, הפעולה מקבלת גם שם טבלה אותו קבעתי בעצמי בקוד בפעולה הראשית app, כפי שאפשר להסתכל:

```
tregB.bind("<ButtonRelease-1>",
lambda event: user_insert(teachers, tfirstName1.get(), tlastName.get(), tvar.get(),
ID.get(), Password.get(), event))
```

קבעתי זאת על מנת להבדיל בין הכנסה של מורה להכנסה של תלמיד.

לאחר מכן הפעולה בודקת מספר בדיקות על מנת לוודא שכל הנתונים תקינים ואין שום פרט חסר. תחילה, היא בודקת האם כל השדות מלאים או שיש שדה אחד או יותר חסרים. במידה ויש היא תשלח הודעת שגיאה שחסר משהו.

לאחר מכן, היא בודקת את תקינות תעודת הזהות, כלומר היא בודקת האם תעודת הזהות מכילה 9 ספרות בלבד.

במידה וכל הנתונים תקינים הפעולה מכניסה את המשתמש לטבלה המתאימה ומסדרת את הטבלה לפי תעודת הזהות של המשתמש.

בנוסף לכך, פעולה חשובה נוספת הבאה לידי מימוש במחלקה זו היא הפונקציה subject_grades. הפונקציה מקבלת מהמשתמש שם של מקצוע, והיא מחזירה לו את כל הציונים שיש לו במקצוע הנתון:

```
def subject_grades(subject_name, gradebox, event):
    if subject_name=="Select Subject":
        messagebox.showerror("Error", "something missing")

    else:
        sub = str(subject_name)
        cursor.execute("SELECT codeID from subjects WHERE subName=?", (sub,))
        subject_row1 = cursor.fetchall()
        for s in subject_row1:
            subjectId = s[0]

            sentence="SELECT subjects.subName,student_grades.grade FROM subjects INNER JOIN
student_grades\
            " ON (subjects.codeID=student_grades.codeId) WHERE student_grades.id== :idinput and
student_grades.codeID== :subcode"
            cursor.execute(sentence, {"idinput": stdid, "subcode": subjectId})

            rows = cursor.fetchall()
            grades_list=list()
            for row in rows:
                grades_list.append(row)
            print("end list")
            return grades_list
```

תחילה הפונקציה בודקת האם המשתמש הזין שם של מקצוע מסוים. במידה ולא היא שולחת הודעת שגיאה.

לאחר מכן, הפעולה הולכת אל טבלת המקצועות ומוציאה ממנה את קוד המקצוע של המקצוע שהתלמיד בחר.

לאחר שסיימה זאת, מתבצעת פעולה הנקראת INNER JOIN שמטרתה לחבר בין 2 טבלאות לפי עמודה משותפת שבמקרה זה היא עמודת `codeId`. הפעולה לוקחת מטבלת `subjects` את שם המקצוע, ומטבלת `student_grades` את הציונים במקצוע זה ומקשרת אותם בטבלה אחרת לפי תעודת הזהות של התלמיד, כך נוצר מצב שבו כל תלמיד מקבל רק את הציון שלו ולא את הציונים של תלמידים אחרים.

לאחר מכן, יצרתי מערך של ציונים שנקרא `grades_list` ואליו הכנסתי את כל הציונים במקצוע הנתון.

בסיום הפעולה היא מחזירה את המערך הנ"ל ופעולה אחרת שנמצאת במחלקת `app` מקבלת את המערך הנ"ל ומדפיסה אותו על גבי ה-`listbox`.

:Calendar

מחלקה זו נמצאת בתוך ספריית utils שיצרתי בפרויקט. המטרה של המחלקה היא לנהל את כל הפעולות הקשורות ללוח המבחנים. המחלקה בנויה מספריית Calendar שבאמצעותה בניתי את לוח השנה, וכן ממספר פעולות שכתבתי אשר הופכות את לוח השנה ללוח מבחנים.

```
def add_test(datePickercalendar,subject,event):
    day=datePickercalendar.day_selected
    month=datePickercalendar.month_selected
    if month<10:
        month=str(0)+str(datePickercalendar.month_selected)
    if day<10:
        day=str(0)+str(datePickercalendar.day_selected)
    year=datePickercalendar.year_selected
    date = str(year) + str(month) + str(day)
    subject=subject
    cursor.execute("SELECT * FROM tests WHERE yyyyymmdd = ?", (date,))

    data = cursor.fetchall()
    if subject=='Select Test':
        messagebox.showerror("Error","Please choose test")
    else:
        if len(data) == 0:
            cursor.execute("INSERT INTO tests(yyyyymmdd,subject)VALUES(?,?)",
                (date,subject))
            messagebox.showinfo("Succeed", "Test was added")
        else:
            messagebox.showerror("Error","couldn't add the test")
    conn.commit()
```

הפעולה add_test מקבלת אליה מספריית Calender את datePickercalendar -פעולה שממנה ניתן לקבל את היום, החודש והשנה שנבחרו בלוח השנה. לאחר מכן היא מבצעת בדיקה האם היום והחודש קטנים מ10 על מנת לדעת האם צריך להוסיף לספרה 0 בעקבות העובדה שהמבנה שבחרתי ללוח השנה הוא yyyyymmdd. לאחר מכן היא בודקת האם המורה בחר מקצוע שבו הוא רוצה לקיים את המבחן, ואם לא שולחת לו הודעת שגיאה. במידה והכל תקין הפעולה מכניסה אל טבלת tests את הנתונים של המבחן שכוללים את התאריך והמקצוע ולאחר מכן הפעולה שולחת הודעה שהמבחן הוכנס בהצלחה. חשוב לציין שהפעולה גם בודקת האם התאריך פנוי ובמידה והוא לא פנוי היא שולחת הודעה שלא ניתן להוסיף את המבחן בתאריך המבוקש.

:Chatserver

מחלקה זו נמצאת בתוך ספריית utils שיצרתי בפרויקט. המטרה של המחלקה היא לנהל את הצ'אט, כלומר מחלקה זו מקבלת ממחלקת chatclient2 הודעות והיא משדרת אותם לשאר הלקוחות המחוברים לצ'אט, כמו כן המחלקה מעדכנת את הלקוחות כאשר לקוח מסוים מתחבר לשיחה וכאשר לקוח מתנתק מהשיחה.

```
def accept_incoming_connections():
    """Sets up handling for incoming clients."""
    while True:
        client, client_address = SERVER.accept()
        print("%s:%s has connected." % client_address)
        client.send(pickle.dumps("Wellcome to Uniapp online chat.\n"
            "Before start chatting please write your name.\n"
            "If you want to quit please write {quit}\n"))
        addresses[client] = client_address
        Thread(target=handle_client, args=(client, client_address)).start()
        print('before checking timeout ...pause')
        # Start checking connection time after at least one joined.
    while NO_CONNECTIONS:
        time.sleep(3)
        Thread(target=checktimeout).start()
```

הפעולה accept_incoming_connections היא הפעולה העיקרית האחראית על תפעול הצ'אט. תחילה הפעולה מקבלת אליה חיבור של לקוח מסוים ומציגה בפניו את הודעת הפתיחה, ולאחר מכן היא מחכה לתגובת הלקוח. בנתיים הפעולה מפעילה thread שקורא לפעולה נוספת handle_client, שאחראית על כל תהליך קבלת ההודעות מהלקוח, כניסה ויציאה מהצ'אט לפי מילת מפתח {quit}. כל תהליך הצגת ההודעות בפני הלקוחות הנוכחים בצ'אט מתרחש בפעולת broadcast. שאחראית רק על ההפצה של ההודעות שהיא מקבלת מפונקציית handle_client.

:Chatclient2

מחלקה זו נמצאת בתוך ספריית utils שיצרתי בפרויקט. המטרה של המחלקה היא לשלוח הודעות לסרבר על מנת שזה ישדר אותם לכל הלקוחות המחוברים לצ'אט. לפני שהפעולה שולחת לסרבר את ההודעות ישנה פעולה חשובה מאוד הנקראת connection שאחראית על החיבור של הלקוח עם השרת.

```
def connection(client_socket, msg_list, top):
    global client_has_disconnected
    try:
        client_socket.connect(('localhost', 33010))
        connected = True
    except:
        print("Server is not up. Please come back later. (1)")
        connected = False

    while connected:
        try:
            data = receive(client_socket, msg_list)
            print("Data received from server: {}".format(data))
        except:
            if client_has_disconnected is False:
                print("Server is not up. Please come back later. (2)")
                break

    top.quit()
    client_socket.close()
```

בתחילת הפעולה הגדרתי משתנה גלובלי שיהיה אחרי לבדיקה האם השרת פעיל בכלל. לאחר מכן הפעולה מנסה להתחבר לשרת לפי ה-IP והפורט שכתובים בו, במידה והיא מצליחה קיים משתנה בוליאני הנקרא connected שכל עוד הוא יהיה שווה true הלקוח יהיה מחובר לשרת ויקבל ממנו מידע חזרה ולאחר מכן יש פעולות של send ו-receive שאחראיות על שליחת הודעות וקבלת הודעות.

סיכום אישי/רפלקציה:

העבודה על הפרויקט הייתה עבורי חוויה שונה ממה שהייתי רגיל אליה עד היום. העבודה דרשה ממני לפתח יכולת לימוד עצמית רבה, במיוחד עקב משבר הקורונה שבעקבותיו כמעט ולא יצא לנו להיפגש עם המורה המלווה ולהיעזר בו. כתוצאה מכך נדרשנו ללמוד בכוחות עצמנו בעזרת האינטרנט, חברים וכו'.

קיבלתי מן העבודה על הפרויקט כלים חשובים מאוד שלדעתי יתרמו לי רבות בעתיד לדוגמא: היכולת ללמוד לבד ולהתמודד עם בעיות גם אם לוקח הרבה זמן כדי לפתור אותם, אני אקח איתי להמשך גם את היכולת לעבוד בצוות מכיוון שללא עזרת חבריי לכיתה לא הייתי מצליח לעשות את הפרויקט והעזרה הייתה הדדית, כאשר גם אני עזרתי לחבריי כאשר נתקלו בבעיה וגם הם עזרו לי. הקשיים שהיו לי בפרויקט בעיקר היו להתמודד עם בעיות מכיוון שבעקבות הקורונה המורה לא היה יכול לעזור באותה רמה כפי שהיה קורה לולא היינו בבית הספר, ולכן נדרשתי להתמודד עם הבעיה בכוחות עצמי, לחפש בעצמי את הבעיה באינטרנט ולמצוא לה פתרונות בכוחות עצמי.

כמו כן, קושי נוסף שהיה לי היה לעמוד בלוחות הזמנים שכן בעקבות משבר הקורונה לוח הזמנים שלנו השתנה והייתי צריך לעבוד שעות רבות בבית על מנת להגיש את הפרויקט בזמן.

במידה והייתי מתחיל את הפרויקט עכשיו הייתי עובד בצורה הרבה יותר מאורגנת ממה שעבדתי בתחילת הפרויקט. הכוונה היא שבתחילת הפרויקט כתבתי את כל שורות הקוד במקום אחד ולא פיצלתי אותם לקטעים קטנים יותר, ולכן כאשר הייתה לי תקלה לא יכולתי למצוא את הבעיה בעקבות העובדה שהקוד היה מבולגן מאוד. לכן, החלטתי להתחיל את הפרויקט מההתחלה ולפרק אותו לגורמים קטנים יותר, פעולה שגרמה לי לאיבוד זמן יקר שהייתי יכול להשקיע אותו בשדרוג הפרויקט ובהוספת רכיבים שרציתי בהתחלה.

לפי דעתי העבודה הייתה יכולה להיות יעילה יותר אם היא הייתה עבודה בצוותים של עד 3 תלמידים, מכיוון שלדעתי עבודה בצוות היא עבודה טובה יותר מכיוון שכל אחד עובד על החלק שלו וכך נכסך זמן רב, וניתן להגיש פרויקטים מושקעים הרבה יותר. בנוסף לכך, עבודה בצוות על פרויקט משותף יכולה לפי דעתי להועיל לתלמידים מכיוון שבעולם ההייטק כמעט על כל פרויקט יש צוות אנשים שעובדים ביחד ולפי דעתי במסגרת תוכנית זו יש הזדמנות להעניק לתלמידים הכנה מוקדמת לדרך שבה אנשים עובדים בעולם העסקי, כך שהם יוכלו להגיע מוכנים יותר ובשלים יותר לעבודה בצבא, ולאחר מכן בעבודה בה יחפצו.

ביבליוגרפיה:

GitHub. (2007, October). Retrieved from <https://github.com/>

Lundh, F. (2009, August). *Events and Bindings*. Retrieved from [effbot.org: https://effbot.org/tkinterbook/tkinter-events-and-bindings.htm](https://effbot.org/effbot.org/tkinterbook/tkinter-events-and-bindings.htm)

McQuistan, A. (2018). *Stack Abuse*. Retrieved from <https://stackabuse.com/a-sqlite-tutorial-with-python/>

Spolsky, J. (2008). *Stackoverflow*. Retrieved from <https://stackoverflow.com/>

Tutorialspoint. (2006). Retrieved from <https://www.tutorialspoint.com/index.htm>

Vishal. (2020, April 26). *PYnative*. Retrieved from <https://pynative.com/python-sqlite-insert-into-table/>